An Improved Approach for Mining Sequential Pattern Based on Tree for Transaction Databases

[1] Priti K. Patel, [2] Smit Thacker
[1] PG Student, [2] Asst. Professor
[1] Department of Computer Engineering,
[1] HJD Institute of Technical Education and Research
Kera, Kutch, Gujarat Technological University, Gujarat, India

Abstract: Data mining is the process of extracting useful information from a large volume of data. Sequential pattern mining is very important technique in the field of data mining. Sequential pattern mining is used to find sequential pattern that occurs in large database. In real world massive amount of data are collected and stored everyday in the databases. Many industries are interested to find useful information in the form of sequential patterns from these databases. Sequential pattern mining is used in various applications such as weblog analysis, DNA sequences, stock market analysis, shopping sequence etc. There are mainly two approaches in sequential pattern mining, first is Apriori based approach and Second is Pattern growth-based approach. This paper introduces a new concept of tree based approach, in which database is scanned only once and all the frequent items and itemssets are arranged as a node of tree, All the frequent items and itemsets are assigned row_id and seq_id. The tree based algortihm used here reduces scan time and minimizes memory overhead.

***IndexTerms* - Sequential pattern mining, frequent pattern, data mining, sequence database.**

I. INTRODUCTION

Data mining is a process of extracting useful information from huge volume of data. It discovers hidden pattern from large volume of data. The sequential pattern mining is a very important concept of data mining, and it is an extension of association rule mining [1].

Sequential pattern mining was first introduced by Agrawal and Srikant in 1995 [2]: "Given a set of sequences, where each sequence consists of a list of elements and each element consists of a set of items and given a user specified min_support threshold, sequential pattern mining is to find all frequent subsequences, i.e., the subsequences whose occurrence frequency in the set of sequences is no less than min_support".

Sequential pattern mining represents relation between different transactions while association rule mining indicates relationship of items in same transaction. Association rule mining finds items that are purchased with each other frequently within same transaction. While sequential pattern mining finds items those are purchased in a unique order by single customer within several transactions. So sequential pattern mining is very useful for a marketing manager to find which item is purchased by particular customer one by one in sequence [3].

II. LITERATURE SURVEY

In [4] author proposed CUSE algorithm. In CUSE algorithm 3D matrix structure is used to extract complete set of frequent sequences. In the first step, all infrequent items of SDB will be eliminated, through a single scan on it using given minsup. In the second step, a 3-dimensional matrix will be constructed by scanning the pruned database in which the dimensions are Sid, items, and number of elements of each sequence, respectively. This 3D matrix is called SequenceCube. CUSE scan database two times and creation of matrix structure requires more memory.

In [5] author introduced CMAP (Co-occurence MAP) structure for storing co-occurrence information. Sequential pattern mining algorithms using a vertical representation are the most efficient for mining sequential patterns in long sequences and have excellent overall performance. The problem of vertical mining algorithms is that they usually spend lot of time evaluating candidates that are infrequent. CMAP is a small and compact structure, which can be built with a single database scan. Author introduced mainly two CMAPs named CMAPi and CMAPs. CMAPi maps each frequent item and CMAPs maps each frequent sequence.

In [6] author introduced algorithm called PrefixSpan-X in order to solve the problem of large space and time overhead in the previous PrefixSpan algorithm. PrefixSpan-X is based on PrefixSpan algorithm. PrefixSpan-X algorithm combine the advantage of prefixspan algorithm and AC automation. The algorithm reduces unnecessary storage space and removes the non-frequent items and combines AC automation to optimize frequent sub-sequence mining, so as to improve the performance. When the support is relatively large, the AC automatically consumes more memory, then memory overhead will be higher than PrefixSpan algorithm.The performance of PrefixSpan-x algorithm is better than Prefixspan algorithm in time and space when the support is low.

In [7] author proposed SWI-GSP(Sliding window interval) algorithm which is based on GSP algorithm. Stock market database is a time series data. In SWI-GSP algorithm stock market data is converted in prepossessed data and then into transaction data table. SWI-

GSP algorithm has better performance than GSP algorithm, because it has interval and window. It can better deal with the time interval of the stock sequence.SWI-GSP algorithm scan database multiple time.

In [8] author proposed Parallel DBP-SPAM algorithm. In previous DBP-SPAM algorithm the main problem is to built PresentIn table is too large when the database is too huge and to find frequent sequences take more time. The proposed algorithm uses separate processing element for block of sequences in DB. Processing element is used to divide the portion of database into equal number of sequences. The Parallel DBP-SPAM is the finest algorithm when the database sequences are very large in size.

In [9] Incremental Sequential PAttern Discovery using Equivalence classes (IncSPADE) algorithm to mine the dynamic database without the requirement of re-scanning the database again. A dynamic database is a database that is frequently updated and increased in term of number of records. IncSPADE is based on SPADE properties to mine updated database without having to start the mining process from the scratch once the database is updated. It is only built prefix lattice tree structure for the newly appended sequences which did not appear previously on the original database. Using this idea, IncSPADE reduce time and memory space needed to mind the original and the updated database.

III. Proposed Method

In this proposed algorithm tree data structure is used for inserting items from transaction database. All the items and itemsets are assigned Row_ID and Seq_ID. This algorithm executes in single scan and tree data structure used here reduces memory overhead.

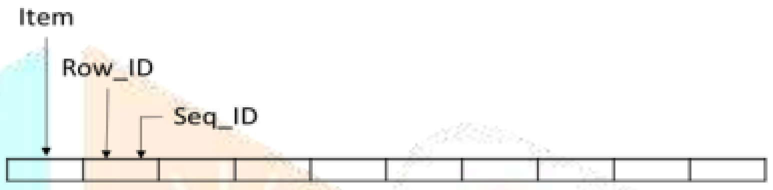


Figure 1 Tree-based structure

IV. Proposed Algorithm

```
Input: Sequence Database (SDB), min_seq_sup
STEP1:
Scan database and add all transaction into tree.
In each transaction, if there is new-item, which is not available in tree, add into tree else store item's Row_ID and Seq_ID into item's node.
//here each node has list of item's Row_ID and Seq_ID
STEP2:
Do for each node
If (length of list of item's node) < min_seq_sup
Remove item's node and re-construct tree.
STEP3:
//Initially X is first item node and Y is next item node.
Do for each item node (X)
      Do for each item node (Y)
If(Row_ID,Seq_ID (X) == Row_ID,Seq_ID (Y))
      count({XY})++;
If(count({XY}) >= min_seq_sup)
{X} U {Y} = {XY}
Add as new item node{XY} into tree.
STEP4:
//Initially X & Y is first item node
Do for each item node(X)
      Do for each item node (Y)
            If (Row_ID(X) == Row_ID(Y) && Seq_ID(X) < Seq_ID(Y))
                  Count({X},{Y})++;
                  If(count({X},{Y}) >= min_seq_sup)
                        Add({X},{Y}) into seqn_freq_patternlist;
                        X=({X},{Y})
```

V. Result Analysis

The experimental results are represented in this section. The proposed approach is implemented using Eclipse Oxygen with Java JDK 1.8 and written in java as a programming language. All the experiments are performed on a core i3 pentium PC with 4 GB main memory, running on Microsoft Windows 10 Pro. Experiments were carried on 4 datasets which are T10I4D100K, Kosarak10k, Bible, and BMS. Table 1 shows execution time of CUSE and Tree based algorithm for different datasets and Table 2 shows memory consumption of CUSE and Tree based algorithm for different datasets.

Table 1 CUSE vs Proposed Tree based approach of Time(ms)

Sr No	Kosarak			T10I4D100K			Bible			BMS		
	Support	Existing	Proposed	Support	Existing	Proposed	Support	Existing	Proposed	Support	Existing	Proposed
1	0.005	5140	1313	0.005	75275	73230	0.03	17000	9782	0.002	5611	3922
2	0.006	3484	984	0.006	68096	66347	0.04	9634	6540	0.003	4273	2984
3	0.007	2396	735	0.007	61205	57756	0.05	6874	4662	0.004	3429	2187
4	0.008	1794	641	0.008	57083	52470	0.06	4403	3387	0.005	3000	1766
5	0.009	1641	578	0.009	48415	46505	0.07	3879	2781	0.006	2547	1487

Table 2 CUSE vs Proposed Tree based approach of Memory(mb)

Sr No	Kosarak			T10I4D100K			Bible			BMS		
	Support	Existing	Proposed	Support	Existing	Proposed	Support	Existing	Proposed	Support	Existing	Proposed
1	0.005	355.48	92.33	0.005	366.10	338.37	0.03	631.34	339.52	0.002	129.10	118.60
2	0.006	181.05	83.14	0.006	299.53	281.28	0.04	287.87	276.70	0.003	105.60	98.03
3	0.007	100.41	79.29	0.007	248.11	224.04	0.05	275.12	254.53	0.004	98.11	94.18
4	0.008	72.28	62.95	0.008	238.12	211.71	0.06	250.41	239.56	0.005	92.62	81.29
5	0.009	66.76	48.76	0.009	233.73	200.88	0.07	246.28	215.53	0.006	56.23	46.59

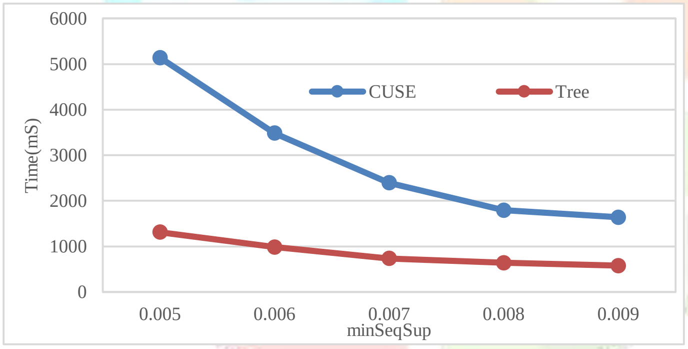


Figure 1 Support vs Time graph of Kosarak

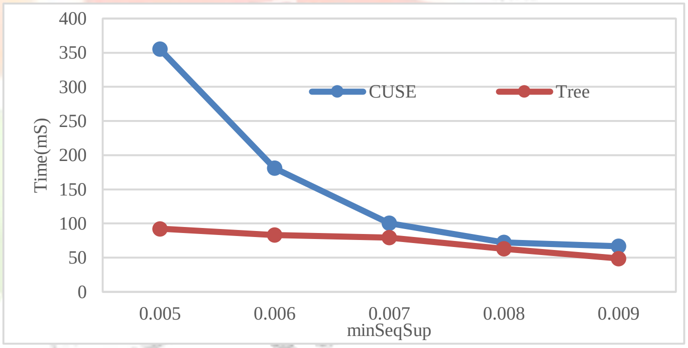


Figure 2 Support vs Memory graph of Kosarak

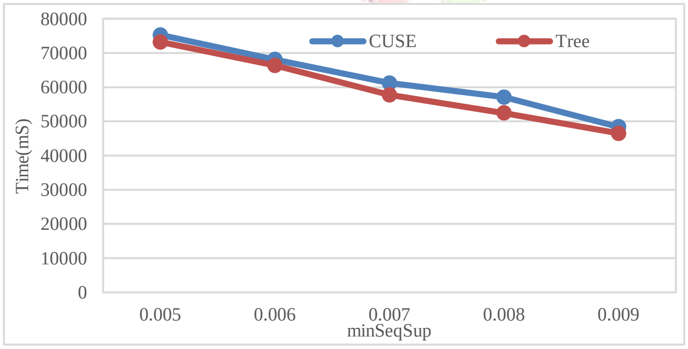


Figure 3 Support vs Time graph of T10I4D100K

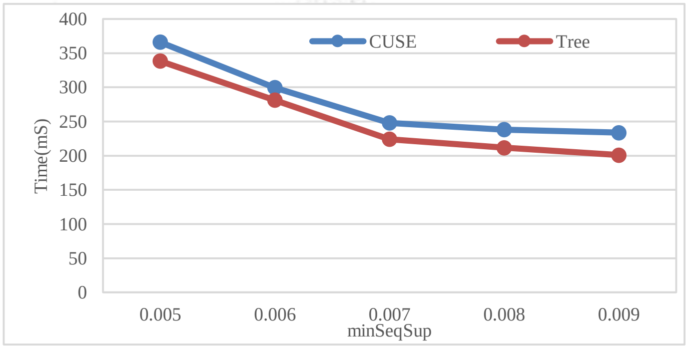


Figure 4 Support vs Memory graph of T10I4D100K

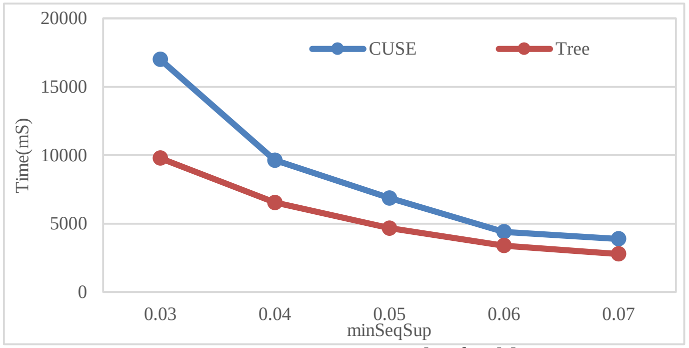


Figure 5 Support vs Time graph of Bible

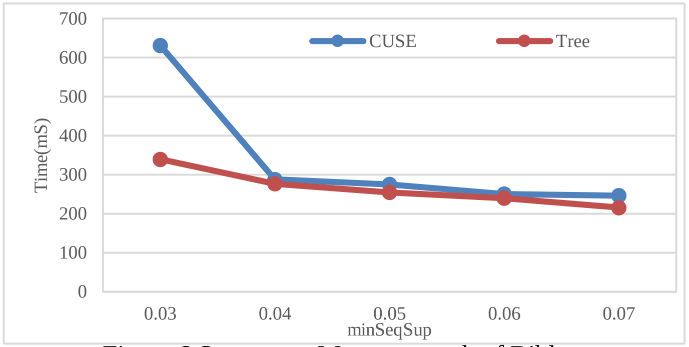
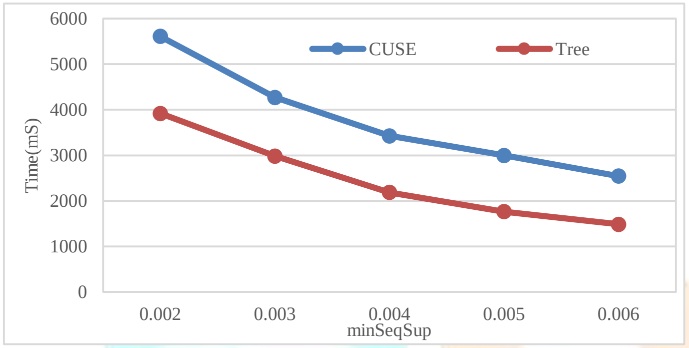


Figure 6 Support vs Memory graph of Bible



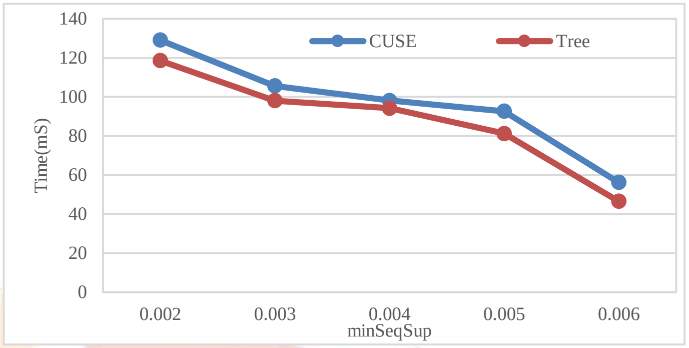


Figure 7 Support vs Time graph of BMS

Figure 8 Support vs Memory graph of BMS

Here Fig 1 shows execution time of CUSE and Tree based algorithm for Kosarak dataset and Fig 2 shows memory consumption of CUSE and Tree based algorithm for Kosarak dataset. Fig 3 and 4 shows execution time and memory consumption graph for T10I4D100K dataset, same way Fig 5 and 6 shows for Bible dataset and Fig 7 and 8 shows for BMS dataset.

VI. Conclusion

In this thesis, we have proposed a tree based approach which can generate the complete set of frequent sequential pattern from a sequence transaction database. This proposed approach reduces the effort of repeated scanning of database due to storage of count in tree's node that help us to enhance the performance of our algorithm. This approach first generate a sequential pattern tree from a sequence transaction database by scanning the database only once which store both frequent and non-frequent items with row_id and seq_id, thus it reduces execution time. Then by using minimum support count we remove non-frequent items from the tree and re-construct the tree. Again, our proposed approach reduces memory consumption by storing only frequent items in tree. The proposed approach experimental results shows that our proposed algorithm performs better than existing CUSE algorithm in terms of memory and execution time.

In future we can modify this algorithm to work on big data. For that , this algorithm can be modified in map reduce programming model.

References

[1] Jiawei Han, Micheline Kamber and Jian Pei, "Data Mining: Concepts and Techniques", Morgan Kaufman publishers is an imprint of Elsevier, 2001

[2] Rakesh Agrawal, and Ramakrishnan Srikant, "Mining sequential patterns". Proceedings of the Eleventh International Conference on Data Engineering, 1995.

[3] M. Chaudhari and C. Mehta, "A Survey on Algorithms for Sequential Pattern Mining", International Journal of Engineering Development and Research, Volume 3, Issue 4, 2015.

[4] Mohammad Karim Sohrabi and Vahid Ghods, "CUSE: A Novel Cube-based Approach for Sequential Pattern Mining", 4th International Symposium on Computational and Business Intelligence 2016.

[5] Philippe Fournier-Viger, Antonio Gomariz, Manuel Campos, and Rincy Thomas, "Fast Vertical Mining of Sequential Patterns Using Co-occurrence Information", Springer International Publishing Switzerland, 2014.

[6] XUE Fei, SHAN Zheng, YAN Li-jing and FAN Chao, "A Improved Sequential Pattern Mining Algorithm Based on PrefixSpan", IEEE, 2016.

[7] Huachun Liu and Hua Du, "Stock Sequence Pattern Mining Method Based on SWI-GSP Algorithm", DMCIT 2017.

[8] K Subramanian, E Elakkiya, "A New Parallel Algorithmic Approach for Sequential Pattern Mining using Binary Representation", IJARCSMS, 2016

[9] Omer Adam, Zailani Abdullah, Amir Ngah, Kasypi Mokhtar, Wan Muhamad Amir Wan Ahmad, Tutut Herawan, Noraziah Ahmad, Mustafa Mat Deris, Abdul Razak Hamdan and Jemal H. Abawajy, "IncSPADE: An Incremental Sequential Pattern Mining Algorithm Based on SPADE Property" Springer International Publishing Switzerland 2016.